PyMol Mutations:
wizard > mutagenesis > protein and then click on your residue, then click the residue you mutate to, then click apply, then file>export molecule and save as a .pdb

Mini Protein dG Calculations

This tutorial is meant to be run from the cluster.

First, we'll source pmx before doing any mutation:

```
'export GMXLIB=~/tutorial_files/pmx/data/mutff45'
```

export GMXLIB=~/pmx/data/mutff45

Then, we'll copy the tutorial files to home directory:

```
`cp -r /zfshomes/nwells/tutorial_files ~`
```

cp -r /home/nwells/tutorial_files ~

And we'll define a script directory environment variable to keep things simple. You will need to do this each time you log into the cluster or else your environment will be reset:

```
`script_dir=~/tutorial_files/scripts`
```

Then we can prepare a directory with two .pdb files: one with W in one state, and the other with W in the other state. We'll call these state_A.pdb and state_B.pdb. Ultimately, we will be calculating the dG of state_A to state_B.

```
`mkdir ~/tutorial_test`
# you can name this whatever you want, just be consistent.

`cp ~/tutorial_files/state*.pdb ~/tutorial_test`
```

Now is when you want to mutate your .pdb files. Ask me for help if you're unsure how to do that.
-I

I recommend you first run it on the 'WT' protein to see if you can replicate the 0 kJ mol result from my simulations.

```
`cd ~/tutorial_test`
```

Then, we'll prepare the systems. The script `multidir_ddg_amber.sh` will create three trials (trial_1-3), each containing a state_A and state_B directory. In each of these state_A and state_B directories, the script will energy-minimize and equilibrate your proteins. At the end, you will end up with a state_A and state_B directory, each containing three directories (/01, /02, and /03), each containing the run input file topol.tpr that we will use to generate our equilibrium trajectories.

From ~/tutorial_test, run the command:

```
`bsub < $script_dir/multidir_ddg_amber.sh`
```

Next, we will run the equilibrium trajectories in /state_A (wt) and /state_B (mut). We will use GROMACS multidir, which runs three simulations per job to better utilize GPU resources. This means one job for all three /state_A simulations, and the same for state_B:

```
`cd ~/tutorial_test/state_A` `bsub < $script_dir/mdrun.sh`

`cd ~/tutorial_test/state_B` `bsub < $script_dir/mdrun.sh`
```

On the exx96 nodes these jobs get around 220 ns/day, so this step should take around 11 hours to run.

The next step is to prepare the morph simulations from our equilibrium trajectories. We first need to put the trajectories in their correct places. From ~/tutorial_test, run:

```
`for i in $(seq 1 3); do rm -r trial_$i/state_A/mdrun
trial_$i/state_B/mdrun; mv state_A/0$i trial_$i/state_A/mdrun;
mv state_B/0$i trial_$i/state_B/mdrun; done`

`for i in $(seq 1 3); do cd trial_$i/state_A/mdrun; cp traj.trr
final.trr ; cd ../../state_B/mdrun; cp traj.trr final.trr ; cd
../../..; done`
```

This next step will submit 3 jobs which generate our morph directories. These jobs will extract one snapshot from each nanosecond of each equilibrium trajectory and place it in its own directory so we can run a morph trajectory on it. Again, run from ~/tutorial_test.

'for i in $(seq 1 3); do rm -r ~/tutorial_test/trial_$i/state_A/morphes; rm -r ~/tutorial_test/trial_$i/state_B/morphes; done' if redo

```
`for i in $(seq 1 3); do cd trial_$i/state_A; bsub <
$script_dir/mutprep.sh ; cd ../state_B; bsub <
$script_dir/mutprep.sh; cd ../..; done`
```

each ~morphes/ will only have frame*.pdb in it after this step

You should now have a directory called /morphes in each of your trial_$x/state_y directories, for a total of 6 /morphes directories. Each morph directory should have 100 frames in it, each with a .pdb file corresponding to that nanosecond of the equilibrium trajectory. To check, you can type `ls trial_*/state*/morphes/*` from ~/tutorial_test.

Once those jobs have finished (a few minutes), we can mutate our snapshots and prepare them for simulation:

```
`for i in $(seq 1 3); do cd trial_$i/state_A/morphes; bsub <
$script_dir/mutrunA.sh ; cd ../../state_B/morphes; bsub <
$script_dir/mutrunB.sh; cd ../../..; done`
```

each ~morphes/ will only have topol.top in it after this step (plus other files)

These scripts call the pmx mutate.py and generate_hybrid_topology.py scripts, as well as some solvation and charge neutralization similar to what's done in prep_amber.sh.

The next step is to run the morphs. This is probably the most time-consuming set of jobs. From ~/tutorial_test, run:

Text

```
`for i in $(seq 1 3); do cd trial_$i/state_A/morphes; bsub <
$script_dir/morphruns.sh; cd ../../..; cd
```

```
trial_$i/state_B/morphes; bsub < $script_dir/morphruns.sh; cd
../../..; done`
```

These jobs will take around 15 hours to complete on the hp12 queue.

The final step is to collect the data and analyze it using pmx and some in-house scritps.

```
`for i in $(seq 1 3); do cd trial_$i;
$script_dir/make_morph_dirs.sh; cd ..; done`

`bsub < $script_dir/dhdl_together.sh`
```

Once this job has finished, you should have an /analysis folder with data corresponding to different “chunks” of the data. For example, /upto50 has data for just the first 50ns of the simulation trajectory.

To analyze this data, you can look at the wplot.png file in /analysis/upto100, which will give you the dG and standard error of the mean for your three trials together. This number corresponds to the delta-G of the W flip for your given protein. Copy this file to valine:

```
`scp analysis/upto100/wplot.png val:`
```